

## Exercise 1: Multiclass and Softmax Regression

- (a) Read in the MNIST data set included in the data sets in the package `keras`, using either

```
library(keras)
mnist <- dataset_mnist()
```

in R or

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist.load_data(
    path='mnist.npz'
)
```

in Python. The data set should contain 4 (2x2) list elements for features (pictures) and outcomes (label) for training and testing (as separate list items).

- (b) Optional: Check that the data is correctly loaded by visualizing it, e.g., like this:

```
## Error in library(keras):  there is no package called 'keras'
## Error in dataset_mnist(): could not find function "dataset_mnist"
## Error in eval(expr, envir, enclos): object 'mnist' not found
## Error in eval(expr, envir, enclos): object 'mnist' not found
## Error in eval(expr, envir, enclos): object 'mnist' not found
## Error in eval(expr, envir, enclos): object 'mnist' not found
## Error in NROW(x_train):  object 'x_train' not found
```

- (c) Convert the features to a (`pandas`) data frame, by flattening the 28x28 images to a 784-entry-long vector, which represents one row in your data frame. Divide the intensity values of each pixel (each column) by 255 to get a value between 0 and 1.

```
## Error in nrow(x_train):  object 'x_train' not found
## Error in nrow(x_test):  object 'x_test' not found
## Error in eval(expr, envir, enclos): object 'x_train' not found
## Error in eval(expr, envir, enclos): object 'x_test' not found
## Error in as.data.frame(x_train): object 'x_train' not found
## Error in as.data.frame(x_test): object 'x_test' not found
```

- (d) Fit a softmax regression to the given data and interpret the results. Use an existing package by searching for a package that fits *Multinomial Logistic Regression*, which is equal to a softmax regression.
- (e) Use the deep learning API `keras` to fit a softmax regression by fitting a network like the following:

```

library(dplyr)
library(keras)

# convert outcome using one-hot encoding
y_train <- to_categorical(y_train)
y_test <- to_categorical(y_test)

neural_network <- keras_model_sequential()

neural_network %>%
  layer_dense(units = ..., # fill in the correct number of units
              activation = ... # fill in the correct activation function
              input_shape = list(784)) %>%
  compile(
    optimizer = "adam",
    loss      = "categorical_crossentropy",
    metric    = "accuracy"
  )

history_minibatches <- fit(
  object      = neural_network,
  x           = as.matrix(x_train),
  y           = y_train,
  batch_size  = 24,
  epochs      = 80,
  validation_data = list(as.matrix(x_test), y_test)
)

```

and compare the learned `neural_network$weights` with the coefficients from the softmax regression.

- (f) Compare the performances of the two algorithms using four different multiclass metrics (you are allowed to use existing implementations).